1. (20 pts.) **Set Cover.**

1. The first selected subset is $\{t,h,r,e,a,d\}$. As all the letters are uncovered elements, we simply pick the set with the most letters (six).
2. The next selected subset is $\{l,o,s,t\}$ because it has three uncovered elements, which is the most compared to other words.
3. The third subset we pick is $\{a,f,r,i,d\}$. While this subset, $\{d,r,a,i,n\}$, and $\{s,h,u,n\}$ all have two uncovered elements i.e. the most currently, $\{a,f,r,i,d\}$ appears first and should be selected based on the tie-breaking rule.
4. The last selected subset is $\{s,h,u,n\}$ since it's the only subset that still has two uncovered elements. After this, all letters/elements are covered.

2. (20 pts.) **Huffman Encoding.**

(a) It's possible to obtain this sequence. Any f_a, f_b, f_c such that $f_a > f_b \geq f_c$ can result in this encoding.

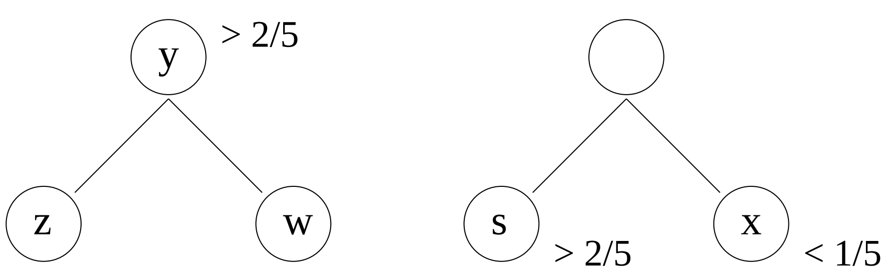
(b) This encoding is not possible since the encoding for a (1), is a prefix of the encoding for c (10).

(c) This encoding also cannot possibly be obtained as it's suboptimal. The encoding for one of the letters in the alphabet could have length 1. For example, the encoding for a could be 0 so that we don't waste space.

3. (20 pts.) **Huffman Properties.**

a Let s be the symbol with the highest frequency (probability) $p(s) > 2/5$ and suppose that it merges with some other symbol during the process of constructing the tree and hence does not correspond to a codeword of length 1. To be merged with some node, the node s and some other node x must be the two with minimum frequencies. This means there was at least one other node y (formed by merging of other nodes), with $p(y) > p(s)$ and $p(y) > p(x)$. Thus, $p(y) > 2/5$ and hence $p(x) < 1/5$.

Now, y must have been formed by merging some two nodes z and w with at least one of them having probability greater than $1/5$ (as they add up to more than 2/5). But this is a contradiction - $p(z)$ and $p(w)$ could not have been the minimum since $p(x) < 1/5$.



b Suppose this is not the case. Let x be a node corresponding to a single character with $p(x) < 1/3$ such that the encoding of x is of length 1. Then x must not merge with any other node till the end. Consider the stage when there are only three leaves - x, y and z left in the tree. At the last stage y, z must merge to form another node so that x still corresponds to a codeword of length 1. But, $p(x) + p(y) + p(z) = 1$

and $p(x) < 1/3$ implies $p(y) + p(z) > 2/3$. Hence, at least one of $p(y)$ or $p(z)$, say $p(z)$, must be greater than $1/3$. But then these two cannot merge since $p(x)$ and $p(y)$ would be the minimum. This leads to a contradiction.

4. (20 pts.) **Feedback Edge Set.** In any connected graph, removing the minimum-weight feedback edge set will leave a spanning tree. This must be the case because a spanning tree has the most edges of any acyclic graph, and a minimum-weight feedback edge set will not remove extra edges after the graph has already become acyclic. So, if G is connected, we can compute the minimum-weight feedback edge set by choosing every edge not in the maximum-weight spanning tree. This requires running Kruskal's algorithm (or Prim's) with every edge weight multiplied by -1 (or equivalently simply modifying the algorithm to choose the heaviest available edge), which has running time $O(|E|\log|V|)$.

If G is not connected, we can repeatedly apply the above approach to construct our final feedback edge set. First, run DFS on G to separate G's edges into connected components in $O(|V|+|E|)$ time. Then, apply the above approach to each connected component. Because E edges are still being considered by Kruskal's algorithm in total, the running time of the entire algorithm is $O(|V|+|E|\log|V|)$. The algorithm is correct because it leaves the heaviest edges that preserve G's connected components while removing all of the lighter edges that make up G's cycles.

Note that Kruskal's Algorithm does not necessarily fail on a disconnected graph. It would create several maximum-weight trees, each of which span a connected component. Therefore, mentioning this fact and running Kruskal's Algorithm a single time is another acceptable solution that runs in $O(|E|\log|V|)$.

5. (20 pts.) **Huffman Efficiency.**

(a) $m\log_2(n) = mk$ bits.

(b) The efficiency is smallest when all characters appear with equal (or near-equal) frequency. In this case, the binary tree that represents the encoding is a complete tree and each encoding takes $\log n = k$ bits. Therefore, encoding the entire file takes mk bits and $E(F) = 1$.

(c) Let F be $x_0, x_1, \ldots, x_{n-2}$ followed by $m-(n-1)$ instances of x_{n-1}. x_{n-1} will be encoded as a single bit, with all of the others being approximately $\log_2(n)+1$ bits (because n is a power of 2, one of the infrequent symbols will be encoded using $\log_2(n)$ bits, but this minor difference will be lost in the big-O notation). This file has efficiency:

$$\frac{m\log_2(n)}{(m-(n-1))\cdot 1 + (n-1)\cdot(\log_2(n)+1)} = \frac{m\log_2(n)}{m-(n-1)+(n-1)\log_2(n)+(n-1)}$$

Assuming m is very large, we have the following big-O notation:

$$= \frac{m\log_2(n)}{O(m)+O(n\log n)} = \frac{m\log_2(n)}{O(m)} = O(\log(n))$$

So the efficiency is $O(\log(n)) = O(k)$

Rubric:

Problem 1, 20 pts

- 3 points: for each correct selected subset.

- 2 points: for correctly identifying the number of uncovered elements in each subset.

Problem 2, 20 pts

(a) 3 points: correct conclusion
3 points: correct set of frequencies

(b) 3 points: correct conclusion
4 points: correct explanation i.e. correctly point out the issue

(c) 3 points: correct conclusion
4 points: correct explanation i.e. correctly point out the issue

Problem 3, 20 pts

(a) 10 pts for right proof

(b) 10 pts for right proof

Problem 4, 20 pts

- 6 points for $E - E'$ is a maximum spanning tree in a connected graph.
- 4 points for how to find the maximum spanning tree.
- 6 points for generalizing to an unconnected graph (Arguing how Kruskal functions properly on the unconnected graph is also valid).
- 4 points for running time analysis.

Problem 5, 20 pts

(a) 6 pts for correct answer.

(b) 6 pts: 3 for $E(f) = 1$, 3 for all frequencies equal.

(c) 8 pts: 4 for correct frequencies, 4 for big-O analysis.